单片微型计算机
原理、接口及应用

徐惠民　安德宁　丁玉珍　编著

DANPIAN WEIXING JISUANJI
YUANLI JIEKOU JI YINGYONG

（第3版）

北京邮电大学出版社
www.buptpress.com

单片微型计算机
原理、接口及应用

（第3版）

徐惠民　安德宁　丁玉珍　编著

北京邮电大学出版社
·北京·

内 容 简 介

本书以 MCS-51 单片机为中心介绍微机原理和接口技术，便于将微机原理的学习和具体的计算机应用实践密切结合。本书从计算机基础知识入手，全面介绍微型计算机的组成、汇编语言程序设计和接口，重点叙述了 MCS-51 单片机的结构、指令系统、程序设计以及对外的接口，包括一些常用接口芯片的使用。相对于第 2 版增加了一章对于 8086 系统的介绍，使得对于微型计算机系统的学习更加完整。

本书可以作为高等院校微机原理或者单片机原理课程的教材，也可以供工程技术人员参考或者作为培训教材。

图书在版编目(CIP)数据

单片微型计算机原理、接口及应用/徐惠民，安德宁，丁玉珍编著．—3 版．—北京：北京邮电大学出版社，2007

ISBN 978-7-5635-1479-3

Ⅰ．单… Ⅱ．①徐…②安…③丁… Ⅲ．单片微型计算机—高等学校—教材 Ⅳ．TP368.1

中国版本图书馆 CIP 数据核字(2007)第 120331 号

书　　名：单片微型计算机原理、接口及应用(第 3 版)
作　　者：徐惠民　安德宁　丁玉珍
责任编辑：郭家宇
出版发行：北京邮电大学出版社
社　　址：北京市海淀区西土城路 10 号(邮编：100876)
北方营销中心：电话：010-62282185　传真：010-62283578
南方营销中心：电话：010-62282902　传真：010-62282735
E-mail：publish@bupt.edu.cn
经　　销：各地新华书店
印　　刷：忠信诚胶印厂
开　　本：787 mm×1 092 mm　1/16
印　　张：22.75
字　　数：536 千字
印　　数：1—5 000 册
版　　次：2007 年 8 月第 3 版　2007 年 8 月第 1 次印刷

ISBN 978-7-5635-1479-3/TP・288　　定　价：29.80 元

前　言

《微机原理》是大学工科各专业的一门计算机硬件技术基础课，特别是通信、电子、控制等专业的重要专业基础课。目的在于加强学生对于微型计算机硬件组成的理解，提高对于计算机硬件的应用能力，甚至硬件开发的能力。它的基本内容包括3个部分：微型计算机组成和工作原理、指令系统和汇编语言程序设计及接口技术。

本书第1版出版于1990年，提出以单片机为基础进行微机原理的课程教学。后来，许多学校的微机原理教学改为以16位处理器为主要内容。但是，仍然有不少学校在使用本书。对于大家的支持与厚爱，我们深表谢意。

十几年来，以MCS-51为核心的单片机芯片一直在各行各业中普遍使用。那种关于8位单片机会很快退出市场的预测并没有被证实。事实表明，在一定的应用领域中，8位单片机的优势还将继续保持下去。

对于微机原理的教学，选择什么样的芯片其实都是可以的，并不一定要限制在一两种处理器中，能够达到教学的要求和培养学生的目的就可以。

使用单片机进行微机原理的教学，比较明显的优点就是比较容易和实践结合起来。学习了单片机的原理和应用后，无论是从课程实验、课程设计，还是学生的创新实践，都很容易找到适当的题目，进行理论和实际结合的训练，从而提高学生的实际应用能力。

但是，8位单片机终究还是有技术上的局限性。为了更加全面地了解微型计算机的发展，学习微型计算机技术，这次修订时增加了关于x86处理器的内容。在这里增加关于x86处理器的内容，主要目的不是要再学习一种微处理器芯片，而是要通过x86处理器补充学习8位单片机还不能覆盖的微型计算机技术，如DMA技术、总线技术等，使得学生有一个更加完整的微型计算机的知识结构。

本次修订中具体增加的内容是仔细选择过的。例如，没有加入8086的指令系统，因为如果没有应用上的需求，没有必要同时学习两种指令系统。但是详细介绍了8086的中断系统，因为它和MCS-51的中断系统刚好在技术上构成互补。

这样的修改，也是和某些学校老师们讨论的结果。感谢这些老师提出的意见和建议。特别感谢成都信息工程学院的杨明欣老师对这本书的修改所作出的建议。也希望听到关于这样修改的更多的反馈。

参加本书编写工作的还有李春宜、邵京婷、龚乃绪、徐晶、葛顺明等。

同时感谢为本书的出版付出辛苦劳动的北京邮电大学出版社的编辑及工作人员。

本书可以作为大学工科各专业计算机硬件技术基础的教材，也可以作为学习计算机硬件基础的培训教材和自学参考书。

书中存在的不足之处，欢迎广大师生和读者批评指正。作者的邮箱地址是 huimin@bupt.edu.cn。

作 者

目　录

第 1 章　微型计算机基础知识

1.1　计算机中负数的表示和运算 …… 1
1.1.1　机器数和真值 …… 1
1.1.2　负数的 3 种表示 …… 3
1.1.3　补码运算 …… 5
1.1.4　原码的乘、除运算 …… 7
1.2　数字电子计算机中的常用编码 …… 9
1.2.1　BCD 码及十进制调整 …… 9
1.2.2　ASCII 码及国内通用字符编码 …… 10
1.3　微型计算机概述 …… 13
1.3.1　微处理器、微型计算机和微型计算机系统 …… 13
1.3.2　微型计算机结构 …… 14
1.3.3　微处理器的基本结构 …… 15
1.3.4　指令执行过程 …… 18
1.4　单片微型计算机 …… 18
1.4.1　单片机的特点 …… 19
1.4.2　单片机的主要品种系列 …… 20
1.4.3　单片机的供应状态 …… 21
习题和思考题 …… 22

第 2 章　微型计算机的存储器

2.1　只读存储器 …… 25
2.1.1　只读存储器的结构及分类 …… 25
2.1.2　只读存储器典型产品举例 …… 28
2.2　随机存取存储器 …… 31
2.2.1　随机存取存储器的基本结构 …… 31
2.2.2　静态基本存储电路 …… 32

2.2.3 动态基本存储电路 …… 33
2.2.4 典型 RAM 芯片举例 …… 34
2.2.5 Flash 存储器 …… 36
2.3 微型计算机存储器的组成与扩展 …… 39
2.3.1 存储器芯片的选择 …… 39
2.3.2 存储器芯片组的连接 …… 40
2.4 CPU 与存储器的接口 …… 44
2.4.1 CPU 与 ROM 的接口 …… 44
2.4.2 CPU 与 RAM 的接口 …… 44
习题和思考题 …… 45

第 3 章 MCS-51 单片机的结构和原理

3.1 MCS-51 系列单片机的结构 …… 47
3.1.1 MCS-51 单片机的基本组成 …… 47
3.1.2 MCS-51 单片机 …… 48
3.1.3 8051 单片机的内部结构 …… 49
3.2 8051 单片机的引脚及其功能 …… 56
3.3 MCS-51 单片机的工作方式 …… 58
3.3.1 复位方式 …… 58
3.3.2 程序执行方式 …… 59
3.3.3 单步执行方式 …… 59
3.3.4 掉电和节电方式 …… 59
3.3.5 编程和校验方式 …… 60
3.4 MCS-51 单片机的时序 …… 61
3.4.1 机器周期和指令周期 …… 61
3.4.2 MC5-51 指令的取指/执行时序 …… 61
3.4.3 访问外部 ROM 和外部 RAM 的时序 …… 62
3.5 MCS-51 单片机外部存储器的扩展 …… 64
3.5.1 程序存储器的扩展 …… 65
3.5.2 数据存储器的扩展 …… 65
3.5.3 单片机和 Flash 存储器的连接 …… 66
习题和思考题 …… 69

第 4 章 MCS-51 单片机的指令系统

4.1 指令和指令程序 …… 71
4.1.1 指令和助记符 …… 71
4.1.2 指令的字节数 …… 72
4.2 寻址方式 …… 73

4.3 数据传送指令…… 79
4.3.1 内部 RAM 单元之间的数据传送指令…… 80
4.3.2 涉及外部存储器的数据传送指令…… 82
4.3.3 堆栈操作指令…… 84
4.3.4 数据交换指令…… 84
4.4 算术运算指令…… 85
4.4.1 加法指令…… 85
4.4.2 带进位加法指令…… 86
4.4.3 加 1 指令…… 87
4.4.4 带借位减法指令和减 1 指令…… 87
4.4.5 乘、除指令 …… 88
4.4.6 十进制调整指令和数据指针加 1 指令…… 89
4.5 逻辑运算及移位指令…… 91
4.5.1 逻辑与运算指令…… 92
4.5.2 逻辑或运算指令…… 92
4.5.3 逻辑异或运算指令…… 92
4.5.4 累加器清零及取反指令…… 93
4.5.5 移位指令…… 94
4.6 控制转移指令…… 96
4.6.1 无条件转移指令…… 96
4.6.2 条件转移指令…… 98
4.6.3 子程序调用及返回指令 …… 101
4.6.4 空操作指令 …… 103
4.7 布尔变量操作指令 …… 103
4.7.1 位传送指令 …… 104
4.7.2 位置位指令 …… 105
4.7.3 位运算指令 …… 105
4.7.4 位控制转移指令 …… 105
习题和思考题…… 108

第 5 章 汇编语言程序设计

5.1 汇编语言源程序的格式 …… 111
5.1.1 标号 …… 112
5.1.2 操作数 …… 112
5.2 伪指令 …… 113
5.2.1 汇编起始命令 ORG …… 113
5.2.2 汇编结束命令 END …… 113
5.2.3 等值命令 EQU …… 114

5.2.4 数据地址赋值命令 DATA …… 114
5.2.5 定义字节指令 DB …… 114
5.2.6 定义字命令 DW …… 115
5.2.7 定义空间命令 DS …… 116
5.2.8 位地址符号命令 BIT …… 116
5.3 汇编语言源程序的人工汇编 …… 116
5.4 MCS-51 程序设计举例 …… 118
5.4.1 顺序程序 …… 118
5.4.2 分支程序 …… 120
5.4.3 循环程序 …… 126
5.4.4 查表程序 …… 130
5.4.5 子程序 …… 133
5.4.6 运算程序 …… 137
习题和思考题 …… 145

第 6 章 微型计算机的输入/输出及中断

6.1 I/O 接口电路概述 …… 148
6.1.1 I/O 接口电路的作用 …… 148
6.1.2 接口与端口的差别 …… 149
6.1.3 外设的编址方式 …… 150
6.1.4 将外设当做数据存储器访问 …… 153
6.2 I/O 传送方式 …… 154
6.2.1 无条件传送方式 …… 154
6.2.2 查询式传送方式 …… 156
6.2.3 中断传送方式 …… 158
6.2.4 直接存储器存取方式 …… 158
6.3 中断概述 …… 159
6.3.1 中断源 …… 160
6.3.2 硬件中断的分类 …… 160
6.3.3 中断的开放与关闭 …… 161
6.3.4 中断源的判别和中断优先级 …… 161
6.4 中断处理过程 …… 163
6.4.1 中断申请 …… 164
6.4.2 中断响应 …… 164
6.4.3 中断处理 …… 165
6.4.4 中断返回 …… 166
6.5 MCS-51 的中断系统及其控制 …… 166
6.5.1 中断系统中的寄存器 …… 166

6.5.2 中断源及中断标志位 …… 166
6.5.3 中断开放的控制 …… 168
6.5.4 中断优先级的控制 …… 169
6.5.5 中断响应 …… 170
6.5.6 中断响应时间 …… 171
6.5.7 中断申请的撤销 …… 172
6.5.8 中断系统初始化 …… 173
6.5.9 中断方式应用举例 …… 174
6.6 MCS-51 外部中断源的扩展 …… 177
6.6.1 借用定时/计数器溢出中断作为外部中断 …… 177
6.6.2 用查询方式扩展中断源 …… 178
习题和思考题 …… 180

第 7 章 MCS-51 的并行接口

7.1 MCS-51 内部 I/O 口及其应用 …… 181
7.1.1 MCS-51 的 I/O 口直接用于输入/输出 …… 181
7.1.2 MCS-51 的 I/O 口改组为非 8 位端口 …… 183
7.2 MCS-51 并行 I/O 口的扩展 …… 186
7.2.1 外接锁存器和缓冲器扩展 I/O 口 …… 186
7.2.2 用 8255A 可编程并行接口芯片扩展 I/O 口 …… 188
7.2.3 用 8155 通用接口芯片扩展 I/O 口 …… 194
7.3 并行口应用——单片机显示/键盘系统 …… 200
7.3.1 LED 数码显示器的控制与编程 …… 200
7.3.2 非编码键盘与单片机的接口 …… 202
7.3.3 显示/键盘系统 …… 206
7.4 MCS-51 内部定时/计数器及其应用 …… 207
7.4.1 工作方式 …… 208
7.4.2 控制方式 …… 209
7.4.3 应用举例 …… 213
7.4.4 电脑时钟 …… 216
7.4.5 复用方式 …… 221
7.5 单片机应用实例 …… 222
7.5.1 液晶显示器 …… 222
7.5.2 车辆计费器 …… 231
习题和思考题 …… 241

第 8 章 单片机与数/模及模/数转换器的接口

8.1 D/A 转换器 …… 243

8.2 MCS-51 单片机与 D/A 转换器的接口 …… 246
8.2.1 DAC0832 D/A 转换器 …… 246
8.2.2 DAC0832 和 MCS-51 单片机的连接 …… 247
8.2.3 8051 单片机和 12 位 D/A 转换器的接口 …… 250
8.2.4 D/A 转换器的应用 …… 252
8.3 A/D 转换器 …… 255
8.3.1 逐次比较型 A/D 转换器 …… 255
8.3.2 双积分型 A/D 转换器 …… 258
8.4 MCS-51 单片机与 A/D 转换器接口 …… 260
8.4.1 ADC0809 A/D 转换器 …… 260
8.4.2 ADC0809 和 8031 的连接 …… 262
8.4.3 对 12 位 A/D 转换器的接口 …… 264
8.5 数据采集和处理系统 …… 266
8.5.1 数据采集和处理系统的硬件 …… 266
8.5.2 数据采集和处理系统的软件 …… 267
习题和思考题 …… 267

第 9 章 MCS-51 系统的串行接口

9.1 串行通信的基本知识 …… 270
9.1.1 串行通信的两种基本方式 …… 270
9.1.2 串行通信中数据的传送方式 …… 272
9.1.3 并/串变换和串行口 …… 273
9.2 MCS-51 单片机的串行口 …… 274
9.2.1 MCS-51 单片机串行口的控制 …… 275
9.2.2 MCS-51 单片机串行口的工作方式 …… 276
9.3 MCS-51 单片机串行口的应用 …… 279
9.3.1 MCS-51 单片机串行通信的波特率 …… 279
9.3.2 串行口方式 0 用做扩展并行 I/O 口 …… 280
9.3.3 串行口方式 1 和方式 3 的发送和接收 …… 282
9.3.4 多机通信 …… 285
9.4 MCS-51 单片机 RS-232 串行口 …… 290
9.5 用 USART 器件扩展 MCS-51 单片机串行口 …… 292
9.5.1 8251A 通用同步/异步接口芯片特性 …… 292
9.5.2 8251A 的结构和引脚功能 …… 292
9.5.3 8251A 的控制字格式 …… 294
9.5.4 8251A 的初始化 …… 295
9.5.5 8251A 和 MCS-51 单片机的连接 …… 296
习题和思考题 …… 297

第 10 章 8086CPU 及 Intel 微机系统

10.1 8086CPU 及个人计算机 …… 298
10.1.1 8086CPU 结构 …… 298
10.1.2 流水线技术的发展 …… 300
10.1.3 8086 的寄存器结构 …… 301
10.1.4 8086 的引脚 …… 303
10.1.5 8086 的工作模式 …… 304
10.1.6 8086 的总线周期 …… 306
10.2 8086 系统扩展 …… 307
10.2.1 8086 和存储器的连接 …… 307
10.2.2 16 位数据的读/写 …… 308
10.2.3 现代微机存储系统的层次结构 …… 309
10.2.4 现代微机的外存储器 …… 311
10.2.5 8086 对外设的访问 …… 314
10.2.6 8086 的中断系统和 8259 中断控制器 …… 315
10.2.7 DMA 过程和 8237DMA 控制器 …… 320
10.3 微型计算机总线的发展 …… 325
10.3.1 不同层次的总线 …… 325
10.3.2 系统总线的主要性能指标 …… 326
10.3.3 微型计算机系统总线的发展 …… 327
10.3.4 USB 总线 …… 330
10.4 微处理器技术的发展 …… 332
10.4.1 Intel 微处理器的发展 …… 332
10.4.2 嵌入式系统微处理器的发展 …… 334
习题和思考题 …… 337
附录 A 测 试 题 …… 339
附录 B MCS-51 系列单片机指令表 …… 344

第 1 章　微型计算机基础知识

这一章介绍微型计算机的基础知识。在内容上与《计算机文化基础》已经介绍过的知识有较大的区别。主要涉及负数在计算机中的表示和运算，微型计算机的基本结构等最基本的知识。

1.1　计算机中负数的表示和运算

在计算机的设计与使用上常使用的数制是二进制、十进制、八进制和十六进制。十进制是人们习惯的数制，但在计算机内部无一例外地都采用二进制。八进制和十六进制只是为了将二进制数表示得更简略时才用到，例如，在汇编语言编程时，经常用十六进制数来表示二进制数。这些关于计算机中数和数值变换的知识是必须掌握的。一般在关于计算机基础的课程中都会介绍，这里不再重复。

但以前所讨论的都是正数的表示和转换。在这一节中则将着重讨论负数在计算机中的表示和运算。

1.1.1　机器数和真值

在计算机中用符号 0、1 表示的数都称为机器数。一个机器数的数值称为它的真值。真值一般用十进制数来表示。

由相同的 0、1 符号构成的机器数并不是只有一种数值。机器数的数值取决于它的构成规则。相同的机器数在不同的规则下，可以有不同的数值。

1. 机器数可以只是正数，也就是无符号数

如果机器数的每一位都是数值位，这样的机器数就只能表示正数，也称为无符号数。

实际上，在计算机基础课程中接触的二进制数只有正数。如 8 位二进制数 10001000 就相当于十进制数 136，而二进制数 01001000 相当于十进制数 72。在这种情况下，所有的二进制数位都有一定的“权”，写出这样的二进制数的“按权展开式”，就可以计算它的真值。

无符号数就是高级语言中的无符号整数(unsigned int)。

可以通过增加无符号数的位数，来扩大无符号数的数值范围。用一个字节表示的无符号数最大的数值是 255；用两个字节表示，最大的数值就是 65 535。

2. 机器数可以用符号位来表示数的正负，就是有符号数

如果机器数要表示正数和负数，则应将数的最高位作为符号位来区分数的正负：最高位为 0 表示正数，最高位为 1 表示负数。这样的机器数也叫有符号数。

有符号数就是高级语言中所定义的整数。

例如，+105 的 8 位机器数应该是 01101001，而 −105 的 8 位机器数则是 11101001。

由于有符号数要用最高位来表示数的正负，使得可以表示的数的范围有所变化：8 位有符号正数的最大值是 01111111，相当于十进制数的 127。

有符号数也可以用两个或多个字节来表示一个有符号数，此时，不论数的字节数有多少，符号位仍定为整个机器数的最高位。

3. 机器数可以用来表示带小数点的数

用机器数来表示带小数点的数通常有两种方法：定点表示法和浮点表示法。在定点表示法中，小数点在数中的位置是固定不变的。对于任意一个二进制数 N 总可以表示为纯整数（或纯小数）和一个 2 的整数次幂的乘积：$N=2^P\times S$。其中 S 称为 N 的尾数，P 称为 N 的阶码，2 称为阶码的底。通常定点表示法中 P 的值是固定的，而在浮点表示法中，P 的值在一定的范围内是可变的。

阶码和尾数都可以是正数，也可以是负数，从而可以表示正的小数或者负的小数。

一般为了扩大数的表示范围，在机器中都采用浮点表示法。这时阶码和尾数要分别表示，并且各自都有自己的符号位，如图 1.1 所示。

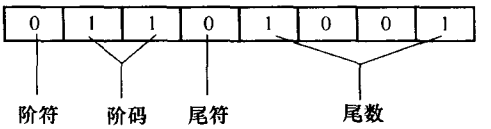

图 1.1　浮点数在机器中的表示

为了尽可能精确地表示一个数，要对尾数进行规格化，就是使尾数是小数点后第一位为 1 的纯小数。然后再来确定阶码的值。

假定用 4 个字节来表示一个浮点数：其中阶码和阶符为一个字节，尾数和尾符为 3 个字节。当然，阶符和尾符各占 1 位。下面用这个浮点数来表示十进制数 256.812 5：

256.812 5 直接表示为二进制数结果是 100000000.1101（注意十进制整数和小数分别转换为二进制数）。尾数规格化的结果是 0.1000000001101。$0.1000000001101\times 2^9$（实际是乘 2^{1001}）才得 100000000.1101。由此可以确定阶码值应该是 9，即二进制数 1001。所以 256.812 5 表示为 4 字节浮点数是 00001001 01000000 00110100 00000000。

现在普遍采用的浮点数格式是 IEEE 制订的浮点数标准。对于单精度浮点数采用 4 个字节，其中，数符占 1 位，指数占 8 位，尾数占 23 位，如图 1.2 所示。

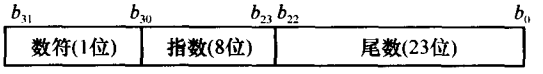

图 1.2　IEEE 浮点数格式

这个标准和上面介绍的浮点数格式主要有两点区别。

(1) 将带小数点的二进制数进行尾数规格化时，应保证在小数点前面有一位整数“1”，但是，写尾数的时候这个“1”是隐含的，不在尾数中出现。例如，256.812 5 对应的二进制数是 100000000.1101，尾数规格化的结果是 1.000000001101，尾数是 000000001101。

(2) 8 位指数是无符号数的形式，对应的十进制数是 0～255，表示阶码－127～＋128。一个具体的阶码，要加 127 后才是 IEEE 浮点数的指数值。例如，100000000.1101 表示为 IEEE 浮点数时，阶码是 8，加 127 后等于 135，即指数是 10000111。

所以，256.812 5 表示为 IEEE 浮点数是：01000011100000000110100000000000。而 11000011100000000110100000000000(C3806800H)的数值就是－256.812 5。

4. 可以通过增加字节数扩大机器数所能表示的数的范围

机器数的数值取决于它表示的是哪一种数。同样是机器数 C3806800H，作为无符号数时的真值是 3 279 972 352；如果是最高位为 1 的有符号数，真值是－1 132 488 704；而作为浮点数时，就是－256.812 5。

所以，不能笼统地说机器数的范围是多少，而是要说机器数所代表的某一种数的范围是多少。4 个字节的机器数，作为无符号数、有符号数和浮点数，所表示的数值的范围是相差很大的。

为了扩大机器数表示的范围，可以增加机器数的字节数。例如，对于 8 位机来说，若用两个字来表示一个无符号数，其数值范围就可以扩大到 65 535。在高级语言程序设计中，某一种数据类型的字节数是固定的。在汇编语言编程时，程序员可以根据需要确定机器数的字节数。例如，可以将有符号数(整数)的长度确定为 8 个字节。

5. 机器数所表示的数值称为机器数的真值

机器数的真值可以用二进制数表示，也可以用十进制数表示。但根据一般的习惯，常用十进制数表示。例如，有符号数 10110110 的真值为－54。

1.1.2　负数的 3 种表示

有符号数在计算机中有若干种表示方法，即为原码、反码和补码。它们共同的特点是都通过符号位来表示数的正负，但是数的大小的表示方法是不同的。

1. 原码

原码的数值部分是用二进制数表示数的绝对值，再在数的前面加一位符号位，符号位为 0 表示正数，符号位为 1 表示负数。

如：$X_1=67$　　$[X_1]_{原}=01000011$

　　$X_2=-67$　　$[X_2]_{原}=11000011$

在原码表示法中 0 有两种表示法，即

$$[+0]_{原}=00000000,\quad [-0]_{原}=10000000$$

原码表示简单易懂，而且与真值的转换方便。但用原码表示的数不便于计算机运算，因为在两个原码数作加法运算时，首先要判断它们的符号，然后再决定用加法还是用减法。若是采用反码或补码表示法，则在两个数的运算中不需要判断数的正负，直接作加法或减法即可，指令的运算比较简单。

2. 反码

一个数的反码很容易求得。如果是正数，它的反码与原码相同；如果是负数，则是它的绝对值连符号位在内按位取反而得到的，即所有的“1”都换成“0”，所有的“0”都换成“1”。也就是说：

若 $X=+x_1x_2\cdots x_n$，则$[X]_{反}=0x_1x_2\cdots x_n$；

若 $X=-x_1x_2\cdots x_n$，则$[X]_{反}=1\,\overline{x_1}\,\overline{x_2}\cdots\overline{x_n}$。

如：$X_1=67=[1000011]_2$　　$[X_1]_{反}=01000011$

$X_2=-67=[-1000011]_2$　　$[X_2]_{反}=10111100$

如果已知一个数的反码，要求它所表示的真值，若是正数则可直接求解，若是负数则可将符号位以外的数值部分按位取反得到负数的原码，然后再求真值。例如：

$$[X]_{反}=01010011,\quad [X]_{原}=01010011,\quad X=+83$$

$$[Y]_{反}=10110011,\quad [Y]_{原}=11001100,\quad Y=-76$$

在反码表示法中，零也有两种形式。对 8 位反码来说：

$$[+0]_{反}=00000000,\qquad [-0]_{反}=11111111$$

在微型计算机中，反码表示法用得较少，因此对于反码的运算也就不作介绍了。

3. 补码

补码是由补数的概念引出来的。一个计量系统所能表示的最大量程被称为模。若模用 K 表示，则当满足

$$Z=nK+Y$$

时，称 Z 和 Y 互为补数。通常 n 取 0，也可以取其他整数。两个互为补数的数，实际上是代表同一个事物。例如，一个圆周角是 360°，在这个圆周系统中，270°和－30°互为补数，因为

$$270°=360°+(-30°)$$

从物理上讲，270°和－30°代表同一个角度。

一个 n 位二进制数 X 的模值为 2^n，因此，X 的补码应为

$$[X]_{补}=2^n+X$$

由于字长 n 位的机器只能表示 n 位数，因此，2^n（它是一个 $n+1$ 位的数“100…0”）在机器中仅能以 n 个 0 来表示。或者说，2^n 和 0 在机器中的表示形式是一样的。

如果将 n 位字长的存数单元的最高位留作符号位，对于正数和负数的补码的求法如下。

当$X=+x_{n-2}x_{n-3}\cdots x_1x_0$ 时，有

$$[X]_{补}=2^n+X=0x_{n-2}x_{n-3}\cdots x_1x_0$$

可见正数的补码也与原码相同。

当$X=-x_{n-2}x_{n-3}\cdots x_1x_0$ 时，有

$$\begin{aligned}[X]_{补}&=2^n+X\\&=2^n-x_{n-2}x_{n-3}\cdots x_1x_0\\&=2^{n-1}+2^{n-1}-x_{n-2}x_{n-3}\cdots x_1x_0\\&=2^{n-1}+(2^{n-1}-1)+1-x_{n-2}x_{n-3}\cdots x_1x_0\end{aligned}$$

$$=2^{n-1}+(11\cdots1-x_{n-2}x_{n-3}\cdots x_1x_0)+1$$
$$=2^{n-1}+\overline{x_{n-2}}\ \overline{x_{n-3}}\cdots\overline{x_1}\ \overline{x_0}+1$$
$$=\overline{0}\ \overline{x_{n-2}}\ \overline{x_{n-3}}\cdots\overline{x_1}\ \overline{x_0}+1$$

所以，负二进制数的补码等于它的绝对值按位取反后加 1，即

$$[X]_{补}=\begin{cases}X & 若\ 2^{n-1}>X\geqslant 0\\ |\overline{X}|+1 & 若\ 0>X\geqslant -2^{n-1}\end{cases}$$

例如，要求 $X=-87$ 的补码，即$[X]_2=-01010111$，则

$$[X]_{补}=\overline{01010111}+1=10101000+1=10101001$$

“求反加 1”需要作两步运算：先对各位求反，再在末位加 1。这个过程也可以简化为一步，即只对负二进制数的各位中最低一位 1 以左的各位求反，而最低一位 1 和右边各位都不变，即可得到负数的补码。例如，要求$[X]_2=-00001000$ 的补码(对应十进制数为−8)，则只需对前面 4 个 0 求反，低 4 位不变，就可得到补码，即$[X]_{补}=11111000$。读者可以自己思考，为什么这个规则是正确的。

补码再次求补以后就可得原数，即真值。对于正数当然没有这样做的必要。对负数的补码经过再次求补并加上负号后，就为原值。例如，对$[X]_{补}=10110110$，则

$$X=-[[X]_{补}]_{补}=-01001010$$

在补码表示法中，0 只有一种表示形式，$[0]_{补}=00\cdots0$。

对于 8 位二进制数来说，用补码所表示的数的范围为−128 ～ +127。

1.1.3　补码运算

在微型计算机中，有符号数一般都以补码的形式在机器中存放和进行运算。用补码对操作数进行运算时不需要先判断操作数的正负，直接做相应的运算就可以，即是加法就做加法运算，是减法就做减法运算。符号位和数值部分一起参加运算，同时也获得结果的符号位和数值部分。

1. 补码加法

不论 X 和 Y 是正数还是负数，可以证明这两个数的和的补码等于两个数补码的和。

$$[X+Y]_{补}=2^n+(X+Y)$$
$$=(2^n+X)+(2^n+Y)$$
$$=[X]_{补}+[Y]_{补}$$

因此，两个数的补码的加法运算可以按以下步骤进行：

- 将两个数先变成补码；
- 对两个补码进行加法运算，若最高位上有进位则舍弃不要；
- 判断结果是否溢出；
- 若结果溢出，则这次运算结果不正确；若没有溢出，对结果再次求补码，得到结果的真值。

所谓溢出是指运算的结果超过了给定长度二进制数可以表示的范围。在加法的情况下，只有两个正数相加或者两个负数相加才有可能出现溢出。判断是否溢出的方法是：若